Accurate Color Filter Array Demosaicking Using High-Order, Direction-Oriented Extrapolation and a Weighted Median Filter

Caroline Houston, Rochester Institute of Technology

***Abstract*—Color-filter-array demosaicking is an essential step in rendering full color images from single-sensor digital cameras. Because each pixel only records one color value, estimates of the missing color information at each pixel must be made. In this paper, extrapolation equations based on higher orders of the Taylor series are proposed for accurate, direction-oriented estimations of missing color information. Four estimations are calculated based on information of pixels in four different directions so that at an edge, there is at least one accurate approximation available for selection as the output. The output estimation is selected using an edge-preserving, weighted-median filter with weights based on a classifying edge orientation map. Because the process avoids interpolation across an edge by using direction-oriented extrapolation, it has been shown that the algorithm performs particularly well even at the edges of objects in an image.**

I. Introduction

Single-sensor digital cameras are coated with a color filter array in order to sample the color information of the scene. Consequently, each pixel only records one of the three primary colors. Estimation of the missing color information, referred to as CFA demosaicking, must be found in order to render a full-color image. The most commonly used CFA is the Bayer pattern illustrated in figure 1. This pattern takes into account the human visual system's increased sensitivity to mid-wavelength (green) light by sampling the green channel at rate twice that of the red and blue channels [1].

Approximation of the missing color values is most challenging at edges in an image due to the fact that there is little correlation between the three channels when moving across the edge of an object [2]. The ideal approach would be to base estimations on pixels along edges instead of across. Therefore, an accurate demosaicking algorithm requires some sort of edge detection/orientation step. Based on this information, an estimation can be made based on the pixels along the edge. It is important to note that even with accurate edge detection, a demosaicking algorithm is limited by the accuracy of the estimation technique.

In this paper, a non-iterative, highly-accurate algorithm is proposed based on extrapolation equations derived from high-order Taylor series approximation. The algorithm consists of two stages, an estimation stage and an edge orientation stage. In the estimation stage, four possible estimations are calculated based on neighboring pixels in four different directions. The equations for these calculations are derived from the Taylor series by including up to the 1st and 2nd order terms for the red/blue and green planes, respectively. In the edge orientation stage, the vertical and horizontal gradients are compared to determine the orientation of a possible edge. Based on this orientation map, weights are assigned to a weigthed-median filter used for selection of the output value. Of the four estimations, the claim is that at least one of them, based on estimations from pixels along the edge, is highly accurate [2]. Section II will present the details and specifics of the proposed technique while Section III will demonstrate the claimed accuracy using simulated and real-world images. Conclusions will be presented in Section IV.

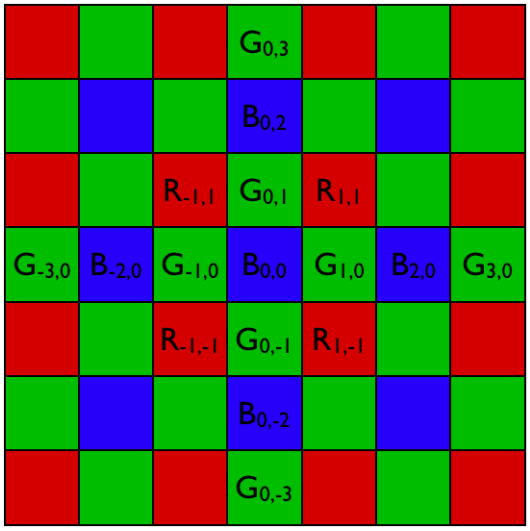


Fig. 1. Bayer Color Filter Array [1]

II. Estimation and Selection Techniques

A. Stage 1 - High Order Extrapolation Using the Taylor Series

In order to use Taylor series approximation you must assume that at some point, the n-th order term and those higher than it can be ignored as they contributes a negligible amount to the final estimation. The basis of high order extrapolation for color filter array demosaicking is that the 1st and 2nd order terms are not negligible and improve the accuracy of estimations in the green plane. The Taylor series is as follows:

$$f(x) = \sum_{m=0}^{\infty} \frac{f^{(m)}(a)}{m!}(x-a)^m \tag{1}$$

Expanding the series we get:

$$f(x) = f(x+1) - f'(x+1) + \frac{1}{2!}f''(x+1) + ... \\ +(-1)^n \frac{1}{n!}f^{(n)}(x+1) + ... \tag{2}$$

To compute the first order approximation we make the assumption:

$$f^{(n)}(x+1) = 0 \text{ for n} \geq 2 \tag{3}$$

Using central difference theorem for first order approximation we arrive at the following equation:

$$f(x) = f(x+1) + \frac{1}{2}(f(x) - f(x+2)) \tag{4}$$

If we consider f(x) to be the value of red/blue/green pixel at a location x, and make the assumption that the pixel values in one plane are well correlated with pixels in the other two planes with constant offsets, we can rewrite the equation for our specific application as follows:

$$\hat{G}_x = G_{x+1} + \frac{1}{2}(B_x - B_{x+2}) \tag{5}$$

Where $\hat{G}_x$ is the estimated green value at x, G_{x+1} is the actual green value one pixel away, B_x is the actual blue value at x, and B_{x+2} is the actual blue value two pixels away.

For more accurate approximation, we include the second order term and make the assumption:

$$f^{(n)}(x+1) = 0 \text{ for n} \geq 3 \tag{6}$$

Following the same methodology described previously, we arrive at a new equation:

$$\hat{G}_x = G_{x+1} + \frac{3}{4}(B_x - B_{x+2}) - \frac{1}{4}(G_{x+1} - G_{x+3}) \tag{7}$$

For the complete and detailed derivation see [3]. This equation produces highly accurate estimations as the results section will demonstrate. Using figure 1 for reference, equations for estimating the green value at a blue pixel in each direction are as follows:

$$\begin{aligned} \hat{G}^T_{0,0} &= G_{0,1} + \frac{3}{4}(B_{0,0} - B_{0,2}) - \frac{1}{4}(G_{0,1} - G_{0,3}) \\ \hat{G}^B_{0,0} &= G_{0,-1} + \frac{3}{4}(B_{0,0} - B_{0,-2}) - \frac{1}{4}(G_{0,-1} - G_{0,-3}) \\ \hat{G}^L_{0,0} &= G_{1,0} + \frac{3}{4}(B_{0,0} - B_{2,0}) - \frac{1}{4}(G_{1,0} - G_{3,0}) \\ \hat{G}^R_{0,0} &= G_{-1,0} + \frac{3}{4}(B_{0,0} - B_{-2,0}) - \frac{1}{4}(G_{-1,0} - G_{-3,0}) \end{aligned} \tag{8}$$

To estimate green at a red pixel, substitute R for B. The important distinction to be made is that the direction-oriented estimations are calculated using pixels on only one side (top, bottom, left, or right). By not using pixels on both sides (top and bottom, left and right), interpolation across edges is avoided. Once the green plane is extrapolated, the red and blue planes are estimated. In most cases, 2nd order approximation is unnecessary as there is only half as much red/blue information as there is green. Therefore, 1st order approximation will suffice [3]. The following equations are used for approximation of the red pixel value at a blue pixel and green pixel, respectively.

$$\begin{aligned} \hat{R}^{TL}_{0,0} &= R_{-1,1} - (\hat{G}_{-1,1} - \hat{G}_{0,0}) \\ \hat{R}^{TR}_{0,0} &= R_{1,1} - (\hat{G}_{1,1} - \hat{G}_{0,0}) \\ \hat{R}^{BL}_{0,0} &= R_{-1,-1} - (\hat{G}_{-1,-1} - \hat{G}_{0,0}) \\ \hat{R}^{BR}_{0,0} &= R_{1,-1} - (\hat{G}_{1,-1} - \hat{G}_{0,0}) \end{aligned} \tag{9}$$

$$\begin{aligned} \hat{R}^{T}_{0,0} &= R_{0,1} - (\hat{G}_{0,1} - G_{0,0}) \\ \hat{R}^{B}_{0,0} &= R_{0,-1} - (\hat{G}_{0,-1} - G_{0,0}) \\ \hat{R}^{R}_{0,0} &= \hat{R}_{1,0} - (\hat{G}_{1,0} - G_{0,0}) \\ \hat{R}^{L}_{0,0} &= \hat{R}_{-1,0} - (\hat{G}_{-1,0} - G_{0,0}) \end{aligned} \tag{10}$$

Extrapolation of the blue plane is done in a similar manner.

B. Stage 2 - Weighted Median Filter Based on Classifier for Selection of Output

Determination of the most accurate estimation calculated in stage one is done using a classifier instead of a linear combiner in order to preserve edges [2]. The classification stage determines the orientation of a possible edge at every pixel by comparing the gradients in orthogonal directions.

When estimating the green color value at a Red/Blue pixel, the vertical and horizontal gradients are defined as follows:

$$V = |G_{0,1} - G_{0,-1}|, \; H = |G_{-1,0} - G_{1,0}| \tag{11}$$

Estimating red/blue at a green pixel:

$$V = |R_{0,1} - R_{0,-1}|, \; H = |B_{-1,0} - B_{1,0}| \tag{12}$$

Estimating red/blue at a blue/red pixel:

$$V = |R_{-1,1} - R_{1,-1}|, \; H = |R_{1,1} - R_{-1,-1}| \tag{13}$$

Estimating red/blue at a blue/red pixel uses diagonal gradients as opposed to vertical/horizontal gradients due to the position of the known red and blue pixels [2]. The gradients at each pixel are compared and a possible vertical edge is indicated with a 1 while a possible horizontal edge is marked with a 0.

$$f(V < H) = \begin{cases} 1, & \text{if } V < H \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

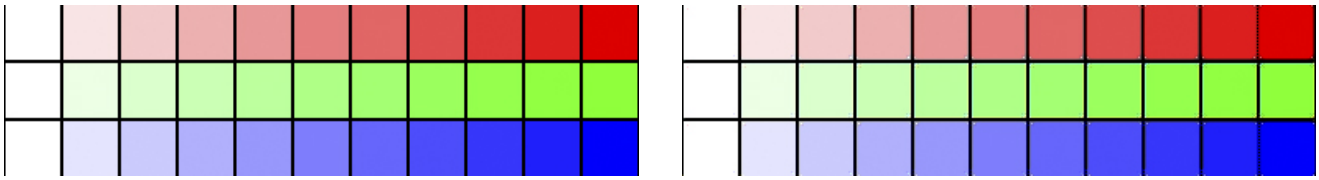
Once the possible orientation of an edge at each pixel is found, weights for the weighted median filter are assigned accordingly. The weighted median filter for this particular application is defined as follows:

$$\hat{G} = \text{MEDIAN}[W_1 \diamond \hat{G^L}, W_2 \diamond \hat{G^R}, W_3 \diamond \hat{G^T}, W_4 \diamond \hat{G^B}] \tag{15}$$

Where W_1, W_2, W_3, W_4 are the weights. The $\diamond$ represents the duplication operator. The duplication operator simply duplicates the value an integer number of times, that is

$$W \diamond X = X, ..., X \tag{16}$$

Where X is the value being duplicated and W is the duplication number.



(a) Original Section (b) Demosaicked Section

Fig. 2.

If a vertical edge is detected at a certain pixel, then the weights for the median filter will be assigned so that the estimations based on pixels in the a vertical direction (i.e. $\hat{G^T}$ and $\hat{G^B}$) are more likely to be selected.

For a possible vertical edge, V<H :

$$\hat{G} = \text{MEDIAN}[\hat{G^L}, \hat{G^R}, 2 \diamond \hat{G^T}, 2 \diamond \hat{G^B}] \quad (17)$$

For a possible horizontal edge, V≥H:

$$\hat{G} = \text{MEDIAN}[2 \diamond \hat{G^L}, 2 \diamond \hat{G^R}, \hat{G^T}, \hat{G^B}] \quad (18)$$

A median filter has the potential to blur edges when there is an even number of values to choose from as there is in the selection process. However, in this 2-D image processing case, the filter will preserve edges due to the fact that there are at least two samples on the same side of an edge. Theoretically, at a vertical edge, the estimations of G^T and G^B should be similar, and the same idea applies for a horizontal edge. If this is indeed the case, the median filter would not blur edges as will be illustrated in the following section [2].

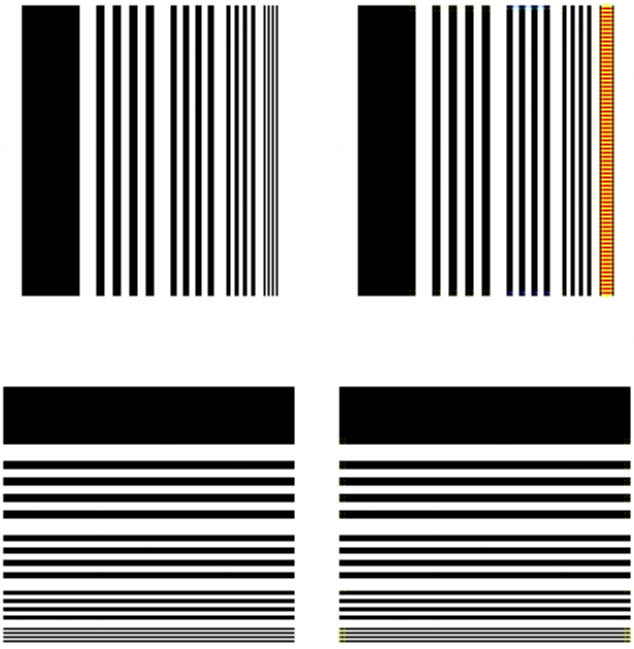
III. Results

The accuracy of this algorithm is measured using two different error calculation methods. Because the most challenging part of CFA demosaicking is estimating missing color information at the edge, the root mean squared deviation at the edge is of particular interest. Calculating the RMSD over the entire image would not be a good indicator of the amount of error as it would be dependent on the amount of high frequency information in the image. Therefore, an RMSD_edge measurement will be used. The measurement is made by using a gradient operation to create and edge mask. The result is then thresholded and dilated and the RMSD for just the edges is found. In addition, the mean color difference ΔE_{ab*} is found over the entire image. A value of 2.3 indicates a just noticable difference in color so a ΔE_{ab*} of 2.3 or less indicates accurate color reporduction. Finally, the error measurements of the demosaiced image will be compared with those of an ideal selector. The ideal selector represents the best possible result the algorithm can produce assuming it is able to correctly select the estimation with the least deviation from the ground truth (original image) at every pixel. The ideal selector is used to confirm the claim that there exits at least one highly accurate estimation out of the four possible.

The first image used for error calculation is an simulated ideal image with a wide range of colors and spatial frequencies. Vertical and horizontal bars spaced as close as one pixel apart are used to show exactly where the algorithm fails and why. An image captured with a modern digital camera most likely would not capture such sharp, high frequency, high contrast edges like those in the simulated image. Therefore, a real-world image will also be used to illustrate the practicality of the algorithm when used on actual scenes captured with a digital camera. In order to compare details, cropped sections of the images are shown. However, the full images (see appendix) are used for the error calculation.

A. Simulated Image

An interesting section of the simulated image is at the corners. Figure 2 illustrates how the accuracy of the algorithm at thin lines, i.e. lines only a few pixels wide, is dependent on their location on the bayer array. Though hard to see in figure 2, the black vertical lines are two pixels wide. Therefore, one of the columns of pixels is on a blue/green row and the other column is on a red/green row. On blue/green rows, red is miscalculated at blue pixels and on red/green rows, blue is miscalculated at red pixels. Another important note is that the miscalculated colors are at the black vertical lines are more

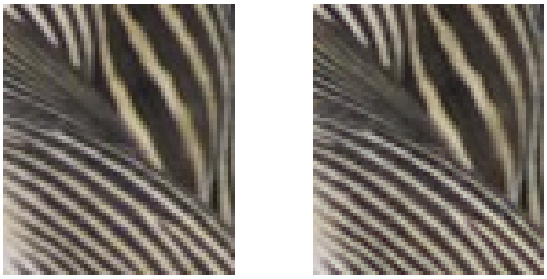


(a) Original Bars (b) Demosaicked Bars

Fig. 3. Performance on Vertical and Horizontal Bars

noticeable as you move towards higher chroma colors (from left to right). Another area of interest is the high frequency horizontal and vertical bars (figure 3). Clearly, the algorithm performs better on the high frequency horizontal bars than the high frequency vertical bars. This is because of the way possible vertical and horizontal edges are found (equation 14). When the vertical gradient equals the horizontal gradient, as is the case at the 1 pixel wide bars, the algorithm declares it a possible horizontal edge even though there is no way to distinguish the orientation. The algorithm could be modified to perform better on vertical bars than on horizontal bars just by changing the inequality in equation 14.

B. Real-World Image



(a) Original Stripes (b) Demosaicked Stripes

Fig. 4. Performance Zebra Stripes

Because a difficult area for the algorithm in the simulated image was the high frequency, high contrast bars, a picture of a zebra was used as the real-world image. The image (see appendix) contains a wide range of spatial frequencies from a low frequency background to the very high frequency stripes shown in figure 4. As you can see, even at the high frequencies, the algorithm performs very well. Imaging systems are limited by many different factors and consequently, are unable to resolve extremely fine details. Towards the center of figurezebra1a, the stripes are no longer resolved. This is advantageous for color demosaicking algorithms as its very difficult to estimate missing information at extremely fine details as demonstrated with the simulated image. The quantitative error evaluation is tabulated below.

TABLE I
ERROR MEASUREMENTS

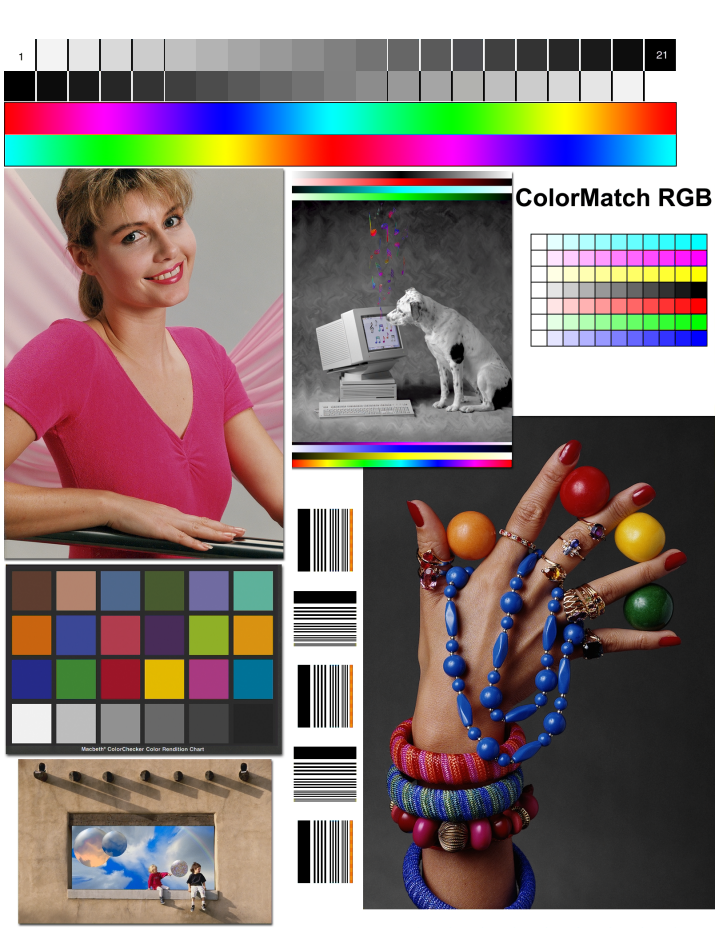
	RMSD_edge (R/G/B channel)	ΔE_{ab*} (mean)
Simulated Image	17.2, 8.95, 20.8	1.69
Ideal Selector (Simulated)	6.61, 3.26, 10.6	0.92
Real-World Image	2.86, 2.91, 3.70	1.32
Ideal Selector (R-W)	1.78, 1.69, 2.32	0.79

Table 1 illustrates how the red and blue channels are less accurate at the edges than the green channel. This can be explained by the amount of known samples we have to work with in the red/blue channels compared to the green channel. Predictably, the error in the real-world image is considerable less than that of the simulated image. In addition, the ΔE_{ab*} values for each image is less than the just noticeable difference of 2.3. Theoretically, this means that on average, the color difference between the original and demosaicked image is not noticeable.

IV. CONCLUSION

High order Taylor series approximation is beneficial in the calculation of direction-oriented estimations of missing color information in digital color images as it will produce at least one highly accurate approximation that can be used for output selection. The use of a weighted-median filter classifier over a linear interpolator preserves edges by selecting the median of window of four values with weights determined by an edge-orientation map. It has been shown that the use of these methods produces highly-accurate results even at high-frequency sections of an image and edges.

APPENDIX



(a) Simulated Image

(b) Real-World Image

Fig. 5. Entire Demosaicked Images Used for Error Calculation

REFERENCES

[1] B. E. Bayer, "Color imaging array," *US Patent 3 971 065*, 1976.

[2] J. S. J. Li and S. Randhawa, "Color filter array demosaicking using high-order interpolation techniques," *IEEE Trans. Image Process.*, vol. 18, no. 9, September 2009.

[3] ——, "Improved accuracy for colour filter array demosaicing using high order extrapolation," in *Proc. 8th Int. Symp. Signal Processing and its Applications*, 2005, pp. 331–334.

[4] ——, "High order extrapolation using taylor series for color filter array demosaicing," in *Springer Lecture Notes in Computer Science*, ser. LNCS 3656. New York: Springer, 2005, pp. 703–711.